CS 1MD3 — Assignment 2

Due Friday, March 14th, 11:59pm

Please read this document very carefully. Follow instructions exactly. If you have any questions please post them to the A2 forum on Avenue.

In addition to basic coding skills, this assignment will focus on loops and lists.

As part of this assignment you will be completing a program that allows a user to play a game of sudoku. You are given a file, `assignment2.py`, with 5 incomplete functions. For this assignment, **you're required to complete these functions**. A description regarding the intended behaviour of these functions is given later in this document. Further documentation and examples for these functions are given in the docstrings within the starter code: `assignment2.py`.

For the purposes of this assignment we will use the following definitions:

The sudoku board type `List[List[int]]` and entries only contain positive numbers. An example of a starting board presented to the player might be the following, where "." indicates an empty cell that needs to be filled by the player:

```
 .  2  3  |  4  .  6  |  7  8  9  |
 .  5  .  |  7  .  .  |  .  .  .  |
 .  8  9  |  .  .  3  |  4  5  .  |
 -  -  -  -  -  -  -  -  -  -  -  -
 2  1  .  |  3  .  .  |  8  .  .  |
 .  6  .  |  8  9  .  |  2  .  .  |
 .  .  7  |  2  1  4  |  .  6  .  |
 -  -  -  -  -  -  -  -  -  -  -  -
 .  3  1  |  6  4  .  |  .  .  8  |
 .  .  .  |  9  7  .  |  5  .  1  |
 9  .  8  |  .  .  1  |  6  4  .  |
```

Empty cells in the board are denoted by 0's within the program:

```
board = [[0,2,3,4,0,6,7,8,9], [0,5,0,7,0,0,0,0,0], [0,8,9,0,0,3,4,5,0],
[2,1,0,3,0,0,8,0,0,], [0,6,0,8,9,0,2,0,0], [0,0,7,2,1,4,0,6,0],
0,3,1,6,4,0,0,0,8], [0,0,0,9,7,0,5,0,1], [9,0,8,0,0,1,6,4,0]]
```

It may be more intuitive to view the board as:

```
board = [[0,2,3,4,0,6,7,8,9],
         [0,5,0,7,0,0,0,0,0],
         [0,8,9,0,0,3,4,5,0],
         [2,1,0,3,0,0,8,0,0],
         [0,6,0,8,9,0,2,0,0],
         [0,0,7,2,1,4,0,6,0],
         [0,3,1,6,4,0,0,0,8],
         [0,0,0,9,7,0,5,0,1],
         [9,0,8,0,0,1,6,4,0]]
```

Note that the board always contains exactly 9 lists, and each list contains 9 indices.

A *cell* is of the type `List[int]` and has a corresponding *x* and *y* coordinate (e.g., cell [0,0] corresponds to the topmost cell on the left, which is the cell at row 0 and column 0). All values in a cell will be between 1 and 9.

The goal of the game is to fill the entire board with numbers 1-9, ensuring that the same number does not repeat within a row, column, or a 3x3 grid. A more thorough explanation of the rules can be found here.

Functions

You are required to implement all of the functions below. Pay attention to parameters of each function, for example, if it is said an input will be a board, you can trust your function will never be tested on input which isn't a board. For further examples of how these functions are intended to operate, view the docstrings of the starter code for this assignment.

1. `print_board(board: List[List[int]]) -> None`
   The parameter is the nested list representing the board. It only prints the board in grid format. It does not return anything.

2. `is_valid(board: List[List[int]], row: int, col: int, num: int) -> bool`
   Receives the current board, as well as the row, column, and number input by the player. Returns True if the row, column, and number combination is valid according to the rules of suduko.

3. `is_complete(board: List[List[int]]) -> bool`
   Receives the current board. Returns True if all cells in the board are filled with the numbers 1-9.

4. `find_empty_cells(board: List[List[int]]) -> List[List[int]]`
   Receives the current board. Returns a list with the positions of all empty cells within the board. Note that the inner lists are always going to be of length 2, containing x-y coordinates.

5. `give_hint(board: List[List[int]], solved_board: List[List[int]]) -> None`
   Receives the current board and a fully solved board (this is created by the code provided to you). It fills an empty cell with the correct answer (if possible) and prints a message to the player. It does not return anything.

Submitting and Grading

This assignment will be submitted electronically via Avenue. Your submitted file must have the name `assignment2.py`, or you may receive 0.

This assignment is worth 10% of your final grade. Grading is done completely automatically. That is, a program calls your function, passes it certain arguments, and checks to see if it returns the expected output. Each function is worth 20% of assignment grade. For any one function, if you pass n of the m tests we run on that function, your grade for that function will be n/m.